```cpp
#include "Game.h"
#include <conio.h>
#include <Windows.h>
#include <iostream>

#include "Enemy.h"
#include "Key.h"
#include "Door.h"
#include "Money.h"
#include "Goal.h"

using namespace std;

constexpr int kArrowInput = 224;
constexpr int kLeftArrow = 75;
constexpr int kRightArrow = 77;
constexpr int kUpArrow = 72;
constexpr int kDownArrow = 80;
constexpr int kEscapeKey = 27;
constexpr int kBackspace = 8;

Game::Game()
    :gameOver{ false } {};

Game::~Game() {};

bool Game::load()
{
    return level1.LoadLevel("Level1.txt", player1.GetXPositionPointer(),
      player1.GetYPositionPointer());
}
void Game::Run()
{
    Draw();
    gameOver = Update();

    if (gameOver)
    {
        Draw();
    }
}

bool Game::isGameOver()
{
    return gameOver;
}

bool Game::Update()
{
    int input = _getch();
    int arrowInput = 0;
    int newPlayerX = player1.GetXPosition();
    int newPlayerY = player1.GetYPosition();

    // One of the Arrow keys were pressed
    if (input == kArrowInput)
    {
        arrowInput = _getch();
    }

    if ((input == kArrowInput && arrowInput == kRightArrow) ||
        ((char)input == 'd' || (char)input == 'D'))
    {
        newPlayerX++;
    }

    if ((input == kArrowInput && arrowInput == kLeftArrow) ||
        ((char)input == 'a' || (char)input == 'A'))
    {
        newPlayerX--;
    }

    if ((input == kArrowInput && arrowInput == kUpArrow) ||
        ((char)input == 'w' || (char)input == 'W'))
    {
        newPlayerY--;
    }

    if ((input == kArrowInput && arrowInput == kDownArrow) ||
        ((char)input == 's' || (char)input == 'S'))
    {
        newPlayerY++;
    }

    if (input == kEscapeKey)
    {
        userQuit = true;
        return true;
    }
    if ((char)input == 'Z' || (char)input == 'z')
    {
        player1.DropKey();
    }

    //If position never changed

    if (newPlayerX == player1.GetXPosition() && newPlayerY ==
      player1.GetYPosition())
    {
        return false;
    }
    else
    {
        return HandleCollision(newPlayerX, newPlayerY);
    }
}
```

```cpp

bool Game::HandleCollision(int newPlayerX, int newPlayerY)
{
    PlaceableActor* collidedActor = level1.UpdateActors(newPlayerX, newPlayerY); // creates a placeable actor
    if (collidedActor != nullptr && collidedActor->IsActive())
    {
        Enemy* collidedEnemy = dynamic_cast<Enemy*>(collidedActor); // specifies the type/ thing we are trying to cast, in this case an enermy
        if (collidedEnemy)
        { // if the pointer is valid, if statement works, if it is a key none of the code will work
            collidedEnemy->Remove(); // if a collision with an enemy occurs, the enermy is removed.
            player1.SetXYPosition(newPlayerX, newPlayerY); // players position is set to new position

            player1.DecrementLives(); // decrmeent lives
            if (player1.GetLive() < 0) // if less than zero game is over.
            {
                return true; // game is over
            }
        }
        Money* collidedMoney = dynamic_cast<Money*>(collidedActor); // if collided with money
        if (collidedMoney)
        {
            collidedMoney->Remove(); // remove the money
            player1.AddMoney(collidedMoney->GetWorth()); // add the money and show the worth.
            player1.SetXYPosition(newPlayerX, newPlayerY);
        }
        Key* collidedKey = dynamic_cast<Key*>(collidedActor);
        if (collidedKey)
        {
            if (!player1.HasKey())
            {
                player1.PickUpKey(collidedKey);
                collidedKey->Remove();
                player1.SetXYPosition(newPlayerX, newPlayerY);
            }
        }
        Door* collidedDoor = dynamic_cast<Door*>(collidedActor);
        if (collidedDoor)
        {
            if (!collidedDoor->IsOpen())
            {
                if (player1.HasKey(collidedDoor->GetColour()))
                {
                    collidedDoor->Open();
                    collidedDoor->Remove();
                    player1.UseKey();
                    player1.SetXYPosition(newPlayerX, newPlayerY);
                }
                else
                {

                }
            }
            else
            {
                player1.SetXYPosition(newPlayerX, newPlayerY); // player goes through the door
            }
        }
        Goal* collidedGoal = dynamic_cast<Goal*>(collidedActor);
        if (collidedGoal)
        {
            collidedGoal->Remove(); // removes actors
            player1.SetXYPosition(newPlayerX, newPlayerY);
            return true;
        }
    }
    else if (level1.IsSpace(newPlayerX, newPlayerY))
    {
        player1.SetXYPosition(newPlayerX, newPlayerY);
    }
    else if (level1.IsWall(newPlayerX, newPlayerY))
    {
        // wall collision
    }
    return false;
}

void Game::Draw()
{
    HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE);
    system("cls");

    level1.Draw();

    //Set cursor position for player
    COORD actorCursorPosition;
    actorCursorPosition.X = player1.GetXPosition();
    actorCursorPosition.Y = player1.GetYPosition();
    SetConsoleCursorPosition(console, actorCursorPosition);
    player1.Draw();


    //Set cursor to end of level.
    COORD currentCursorPosition;
    actorCursorPosition.X = 0;
    actorCursorPosition.Y = level1.GetHeight();
    SetConsoleCursorPosition(console, actorCursorPosition);
}
```